Identificación:

- Andrés Felipe Rojas Pinzón - 201715859
- Ramón Alejandro Arias Rivera - 201718714

Algoritmo de solución:

- Pre-condición: La entrada “n” que entra por parámetro concuerda con el tamaño de los arreglos, C que es el capital de Juan es un número entero mayor a cero, los números que entran por parámetro en cada arreglo con números enteros.
- Post-Condición: El capital retornado es el máximo que se puede lograr invirtiendo correctamente en las bolsas A y B.
- Invariante: En cada iteración, la inversión que se hace en la bolsa A o en la bolsa B es siempre la mejor ya que se verifica esto mirando los cuatro siguientes números (los números de la posibilidad de inversión en el tiempo `k+1` y en `k+2`) si `k <= n-3` o mirando la posibilidad en `k+1` cuando `k == n-2`.
- Solución: Este problema se podía solucionar con grafos ya que era un problema de decisión o con programación dinámica. Sin embargo, ya que era un problema con decisiones sencilla se optó por un solo recorrido de la dos listas. Durante el desarrollo del problema se volvió evidente que solo era necesario pronosticar las posibilidades de inversión de las siguientes dos iteraciones si `k <= n-3` o si la siguiente iteración si `k == n-2`, de esta manera siempre se eligió la mejor decisión de inversión en el tiempo `k` dependiendo de los valores de las siguientes posibilidades. Se escogió esta implementación porque, a pesar de la cantidad de líneas y condicionales, se lograba una complejidad temporal baja (O(n)) y una complejidad espacial constante ya que no usa otra estructura de datos aparte de las que entra por entrada.

Análisis de complejidades espacial y temporal:

- La complejidad temporal es lineal ya que solo se recorre el tamaño de los arreglos una vez. (O(n))
- La complejidad espacial es constante ya que no se usa otra estructura de datos aparte para almacenar información, solo la que entra por parámetro. (O(1)).

Comentarios finales:

- Es bastante probable que el problema se pudiera solucionar con menos líneas de código, sin embargo la solución propuesta logra una complejidad temporal bastante baja al igual que la complejidad espacial, así que, si es cierto que las líneas de código y cantidad de condicionales no fueron los óptimos, esto se compensa con un rendimiento alto del algoritmo.